

Intro. to machine learning (CSI 5325)

Lecture 27: Boosting

Greg Hamerly

Spring 2008

- 1 Ensemble methods
- 2 Boosting fundamentals
- 3 AdaBoost
- 4 AdaBoost analysis

Ensemble methods

Ensemble: using many classifiers together

- could be different types of classifiers
- each classifier gives a different view
- usually each is trained differently
- need a sensible method of combining classifiers

Popular methods:

- bagging – learning over different resamples
- cross-validation – usually models aren't combined
- boosting – learning over different distributions

Combining classifiers

Suppose that we have:

- n labeled examples (x_i, y_i) , where $y_i \in \{-1, +1\}$
- a number of T hypotheses h_1, h_2, \dots, h_T , each trained on the data

How should we combine the T classifiers?

- choose h_t with the least training error

$$h(x) = h_m(x) \text{ where } m = \underset{t}{\operatorname{argmax}} \sum_{i=1}^n y_i h_t(x_i)$$

- choose h_t with the least validation error
- use all T classifiers in a sum

$$h(x) = \operatorname{sign} \left(\sum_{t=1}^T h_t(x) \right)$$

Combining classifiers (with weights)

Suppose that we have:

- n labeled examples (x_i, y_i) , where $y_i \in \{-1, +1\}$
- a number of T hypotheses h_1, h_2, \dots, h_T , each trained on the data
- **a weight α_t for each hypothesis h_t** , which indicates its relative quality

Then we could form:

$$h(x) = \text{sign} \left(\sum_{t=1}^T \alpha_t h_t(x) \right)$$

This is the AdaBoost classifier.

Ideas behind boosting

- we have a number of 'weak' hypotheses h_t (sometimes called weak learners)
- each hypothesis is good enough to be right most of the time, but not necessarily better than that
- think of these as 'rules of thumb'

We want to combine a number of weak hypotheses to get better classification performance.

What is a weak learner?

A weak learner is one which is expected to get $< 50\%$ error rate (test rate) on any binary learning task.

50% is usually an upper limit – why?

- if the hypothesis is wrong more than this, just reverse its decisions

This is an *expectation* of error over random datasets.

- Certainly on some datasets a learner may have poor generalization, due to misleading training data.

Start with a class of hypotheses

Choose a learner which can produce a weak hypothesis:

- decision trees – which are not very weak
- decision stumps – one-node decision trees
- hyperplane classifiers

Domain knowledge can help here.

Where do we get the weak hypotheses?

Different hypotheses come from different learning tasks.

- 'what would you do on this dataset?'
- 'what would you do on that dataset?'
- ...

However, we only have one dataset (usually).

AdaBoost (the most popular boosting algorithm) re-uses the same dataset to learn different hypotheses.

The original dataset does not change, but the importance of each example does.

AdaBoost algorithm (Freund & Schapire)

Initialize $D_1(i) = 1/n$

For $t = 1, \dots, T$:

- train weak learner using distribution D_t to get hypothesis h_t
- $h_t : X \rightarrow \{-1, +1\}$ has error $\varepsilon_t = Pr_{i \sim D_t}[h_t(x_i) \neq y_i]$
- let $\alpha_t = \frac{1}{2} \log \left(\frac{1-\varepsilon_t}{\varepsilon_t} \right)$
- update the distribution:

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\ e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i \end{cases}$$

where Z_t is a normalizing term so D_{t+1} is a distribution.

$$\text{Classifier: } H(x) = \text{sign} \left(\sum_{t=1}^T \alpha_t h_t(x) \right)$$

Error of a weak hypothesis

The error of hypothesis h_t is with respect to the distribution D_t :

$$\varepsilon_t = \Pr_{i \sim D_t}[h_t(x_i) \neq y_i]$$

Compute as:

$$\varepsilon_t = \sum_{i=1}^n D_t(i) I(h_t(x_i) \neq y_i)$$

If $D_t(i) = 0$, then we don't care about how h_t classifies it.

Each h_t should minimize the weighted error ε_t , which as opposed to the assumption that all errors are equally bad.

Incorporating D_t weights

If a weak learner can incorporate weights directly, then just use that:

- decision trees/stumps
- neural networks w/backpropagation

If a weak learner cannot incorporate weights, then resample the data according to D_t .

Behavior of the weights

If an example is misclassified, its weight increases. Otherwise its weight decreases.

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\ e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i \end{cases}$$

If an example is always misclassified, then it becomes much more important, since the weights are retained over time.

How much can the weights change?

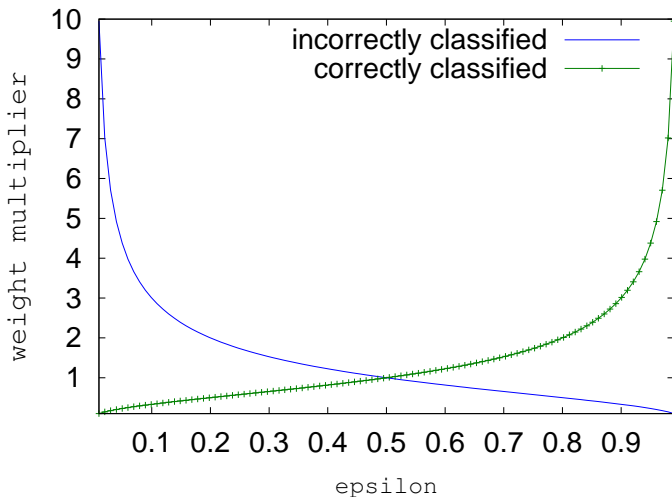
$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\ e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i \end{cases}$$

recall:

- $0 \leq \varepsilon_t \leq 1$
- $\alpha_t = \frac{1}{2} \log \left(\frac{1-\varepsilon_t}{\varepsilon_t} \right)$
- $e^{\alpha_t} = \sqrt{(1-\varepsilon_t)/\varepsilon_t}$
- $e^{-\alpha_t} = \sqrt{\varepsilon_t/(1-\varepsilon_t)}$

Thus, if $\varepsilon_t \approx 0$, then $D_t(i) \rightarrow 0$

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} \sqrt{\varepsilon_t/(1-\varepsilon_t)} & \text{if } h_t(x_i) = y_i \\ \sqrt{(1-\varepsilon_t)/\varepsilon_t} & \text{if } h_t(x_i) \neq y_i \end{cases}$$



Role of α_t

Recall the final classifier's form:

$$H(x) = \text{sign} \left(\sum_{t=1}^T \alpha_t h_t(x) \right)$$

Also,

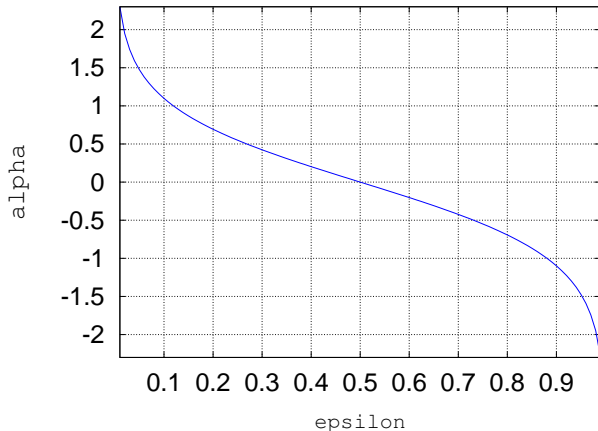
$$\alpha_t = \frac{1}{2} \log \left(\frac{1 - \varepsilon_t}{\varepsilon_t} \right)$$

Thus:

- $\varepsilon_t = 0.5 \rightarrow \alpha_t = 0$
- $\varepsilon_t < 0.5 \rightarrow \alpha_t > 0$
- $\varepsilon_t > 0.5 \rightarrow \alpha_t < 0$

Relationship of α_t and ε_t

$$\alpha_t = \frac{1}{2} \log \left(\frac{1 - \varepsilon_t}{\varepsilon_t} \right)$$



Greedy search

Interestingly, AdaBoost is a greedy strategy.

It always minimizes the error with respect to the current D_t , and does not backtrack.

In this way, it's like decision tree learning.

How does boosting help?

- focuses effort on hard-to-classify examples
- takes hard work off the learner
- learners only have to specialize in small areas
- reduces classifier bias
 - simple hypothesis usually has high bias
 - ensemble can make more complicated decisions and lower bias
- reduces classifier variance (perhaps)
- can identify outliers

Boosting works well when...

- the margin between the two classes is large (as in SVMs)
- the data is fairly noise-free
- it's easy to discover simple rules
- the weak learner can take a distribution over examples

Difficulties with boosting

- noisy data may cause overfitting
- many rounds of boosting may be costly (for training & test)
- ensemble methods are all typically hard to understand

Training error

Training error of the final H is bounded by:

$$\prod_t 2\sqrt{\varepsilon_t(1-\varepsilon_t)} \leq \exp\left(-2\sum_t \left(\frac{1}{2} - \varepsilon_t\right)^2\right)$$

Thus, training error goes down exponentially with more rounds of boosting.

Generalization error

Generalization error of the final H is bounded by:

$$\hat{Pr}(H(x) \neq y) + O\left(\sqrt{Td/n}\right)$$

(with high probability).

Terms:

- $\hat{Pr}(H(x) \neq y)$ is the training error
- T is the number of boosting rounds
- d is the VC-dimension of the weak learner
- n is the number of examples

Generalization error (2)

Generalization error of the final H is bounded by:

$$\hat{Pr}(\text{margin}(x, y) \leq \theta) + O\left(\sqrt{d/(n\theta^2)}\right)$$

where the margin of (x, y) is

$$\frac{y \sum_t \alpha_t h_t(x)}{\sum_t \alpha_t}$$

Claim: this bound is independent of the number of boosting rounds.

- though T isn't in the bound, for lower d , more boosting rounds are likely necessary to obtain a larger margin θ ...

Improving generalization with more boosting

Many have observed that even after training error is zero, generalization error may continue to improve with more boosting.